electronics

MDPI

Article

A Genetic Optimized Federated Learning Approach for Joint Consideration of End-to-End Delay and Data Privacy in Vehicular Networks

Müge Erel-Özçevik, Akın Özçift, Yusuf Özçevik and Fatih Yücalar *

Department of Software Engineering, Manisa Celal Bayar University, 45400 Manisa, Turkey; muge.ozcevik@cbu.edu.tr (M.E.-Ö.); akin.ozcift@cbu.edu.tr (A.Ö.); yusuf.ozcevik@cbu.edu.tr (Y.Ö.)
* Correspondence: fatih.yucalar@cbu.edu.tr

Abstract: In 5G vehicular networks, two key challenges have become apparent, including end-to-end delay minimization and data privacy. Learning-based approaches have been used to alleviate these, either by predicting delay or protecting privacy. Traditional approaches train machine learning models on local devices or cloud servers, each with their own trade-offs. While pure-federated learning protects privacy, it sacrifices delay prediction performance. In contrast, centralized training improves delay prediction but violates privacy. Existing studies in the literature overlook the effect of training location on delay prediction and data privacy. To address both issues, we propose a novel genetic algorithm optimized federated learning (GAoFL) approach in which end-to-end delay prediction and data privacy are jointly considered to obtain an optimal solution. For this purpose, we analytically define a novel end-to-end delay formula and data privacy metrics. Accordingly, a novel fitness function is formulated to optimize both the location of training model and data privacy. In conclusion, according to the evaluation results, it can be advocated that the outcomes of the study highlight that training location significantly affects privacy and performance. Moreover, it can be claimed that the proposed GAoFL improves data privacy compared to centralized learning while achieving better delay prediction than other federated methods, offering a valuable solution for 5G vehicular computing.

Keywords: cloud-based vehicular technologies; genetic algorithm; machine learning; security; vehicular networks

Citation: Erel-Özçevik, M.; Özçift, A.; Özçevik, Y.; Yücalar, F. A Genetic Optimized Federated Learning Approach for Joint Consideration of End-to-End Delay and Data Privacy in Vehicular Networks. *Electronics* **2024**, *13*, 4261. https://doi.org/10.3390/electronics13214261

Academic Editor: Yolanda Blanco Fernández

Received: 26 September 2024
Revised: 24 October 2024
Accepted: 28 October 2024
Published: 30 October 2024

1. Introduction

In intelligent transport systems, there are many applications that require vehicular sensing. Data collection is performed by vehicular sensors for edge, fog, and cloud computing. Here, road side units (RSUs) have an active role in fog computing and sharing the vehicular data with edge devices and the cloud [1]. In particular, end-to-end delay; i.e., from cloud to vehicle in 5G vehicular networks, should be under a few milliseconds [2,3]. Given that vehicular data collectively consume over 70% of the physical resources on the edge, vehicular cloud computing can be an alternative solution, but it is imperative to develop policies addressing acceptable delay requirements [4,5].

Moreover, privacy-preserving data computing is another problem in vehicular computing [6]. The privacy of the vehicular data provider should be preserved while collecting data from vehicular sensors or end-user devices from the edge to the cloud to improve safety in vehicular networks. Therefore, the problem can be defined as (1) minimizing end-to-end delay for acceptable vehicular performance and (2) preserving data privacy for vehicular networks. There are many studies that try to consider each of them in vehicular networks. For example, in [7], the cost-effective task offloading is optimized by using the Stackelberg game approach , whereas in [1], reliable data sharing is provided by using blockchain, and the study in [8] proposes another approach for an indistinguishable

privacy-preserving model for vehicular cloud computing. Nevertheless, most of the solutions proposed in the literature focus on either minimal delay prediction or data privacy. Hence, we revise the corresponding literature from the underlined perspectives.

1.1. Related Works

In particular, vehicular network traffic has dynamic characteristics; therefore, the prediction problems/solutions in the 5G vehicular network domain should be learned from streaming data [9,10]. Nowadays, machine-learning-based approaches are candidate solutions for dynamic delay prediction to minimize end-to-end delay [11]. Then, our literature survey will focus on machine learning approaches in terms of end-to-end delay prediction and data privacy perspectives. Since there is no study that considers both end-to-end delay prediction and user data privacy by optimizing the task computing for vehicular networks, to the best of our knowledge, we will handle the literature in these two categories separately:

i. End-to-end delay prediction: In [12], the authors proposed Online Mobile Edge Data Caching (OL-MEDC), formulating mobile edge caching to predict delay effectively. In [13], end-to-end delay is considered as a network bottleneck and as a negative component in communication delay. The study handles delay prediction as a long-term time-varying optimization problem and solves it using a cost-effective approach to balance access delay, communication delay, and service switching cost. For instance, the study of [14] claims that computing tasks on edge devices suffer in efficiency due to the limited capacity on edge devices. On the other side, there is a trend to train tasks on edge networks to overcome the latency problem while carrying the trained model from the cloud to the vehicle device. In [15], a machine-learning-based solution is proposed to keep minimal latency requirement, and the authors claim that this is the only way for the scenarios that have too much data and too many parameters.

ii. Data privacy: For preserving data privacy, in [16], a distributed deep learning algorithm is proposed for popularity prediction in mobile edge computing. Although it solves data deficiency resulting from data privacy, it does not consider the delay observed by model training in both vehicles, RSU servers, and centralized cloud servers. To preserve data privacy, federated learning is defined as a secure learning model in the vehicular literature [17–19]. It has a recursive running principle between end-user devices and cloud servers. In each iteration, end-user devices train their own model by using limited data. Without sharing data publicly, it shares only the trained model with the cloud. Thanks to this approach, it keeps the data privacy of the end-user. Afterwards, the cloud server aggregates the trained models and regenerates a new model for end-users. In the next iteration, by taking the recent model, end-user devices keep training a new model on the dynamically changed data. However, federated learning does not completely guarantee the protection of client privacy [20]. Therefore, the model transferring between servers is a significant challenge while protecting vehicle privacy. Especially, the data leakage attack is a significant problem while sharing the model parameters between servers. More specifically, when a vehicle is attacked, the model parameters transferred to the RSU/cloud server are not reliable and the performance of federated learning may be negatively affected. In this study, we aim to keep the model performance at the highest level by running our federated learning method with differential privacy algorithms given by [21–23] that minimize the possibility of leakage attacks. On the other hand, while ensuring data privacy, the total model training process should be completed as quickly as possible to assure 5G requirements. Thus, some research questions should be solved by astute strategies to respond as to whether the model should remain in the local vehicle, whether the model parameters should be shared by the RSUs or with the cloud server, and whether the model should be retrained by the RSUs or the cloud server. In [21–23], the authors highlight the advantages of federated learning in vehicular scenarios for privacy and performance. However, optimizing federated learning for latency and model accuracy remains a trade-off problem to be attentively solved.

The communication overhead between end-user devices and the cloud server during the model aggregation also causes complexity [24]. In [25], the communication complexities between distributed and centralized learning approaches are analytically analyzed. To meet high prediction performance, the authors proposed to aggregate trained models from end-user devices to the cloud server. Therefore, the distributed approaches have more communication complexity than other approaches. Although pure federated learning in distributed end-user devices has less processing delay, it lacks end-to-end delay prediction performance due to limited data, and this brings out communication complexity between the cloud server and the end-user/vehicular devices [26–28].

As is observed from the literature survey, the studies do not consider the location of model training to determine its effect on the delay performance and data privacy. Since there is no consensus on the solutions and the combination of these problems, a preliminary analysis is needed to determine the relation between end-to-end delay and data privacy while considering the optimal location of model training.

1.2. Preliminary Analysis

As we mentioned in the problem definition, end-to-end delay, which includes model training and processing times in servers, and data privacy of end-users are vital in vehicular networks [29]. In other words, from the end-user perspective, they should both be satisfied. However, to the best of our knowledge, there is no study that considers the trade-off between delay and data privacy on complex and dynamic scenarios in the literature. Therefore, we execute preliminary analysis for delay and data privacy while computing tasks in between vehicles, road side units (RSUs), and cloud servers, as shown in Figure 1. During the analysis, the same delays have been observed at different locations of task computing. For example, case (a), which runs 23% of tasks on RSU servers and 77% of tasks on the centralized cloud server, and case (b), which runs 29% of tasks on vehicle and 71% of tasks on the centralized cloud server, result in the same delay which is detailed in Section 3. Data privacy is more preserved if the tasks are mostly computed on the vehicle. As preliminary analysis shows, case (a) does not run any task on the vehicle, but case (b) runs 29% of tasks on the vehicle device. On the other hand, the probability of data leakage attack is higher in case (a) than case (b) according to the privacy-preserving perspective introduced in [21–23]. Though these two cases obtain the same end-to-end delay, case (b) maintains the privacy of vehicle data compared to case (a). Hence, we propose a joint consideration of end-to-end delay and data privacy while computing tasks in vehicles, RSUs, and the cloud.

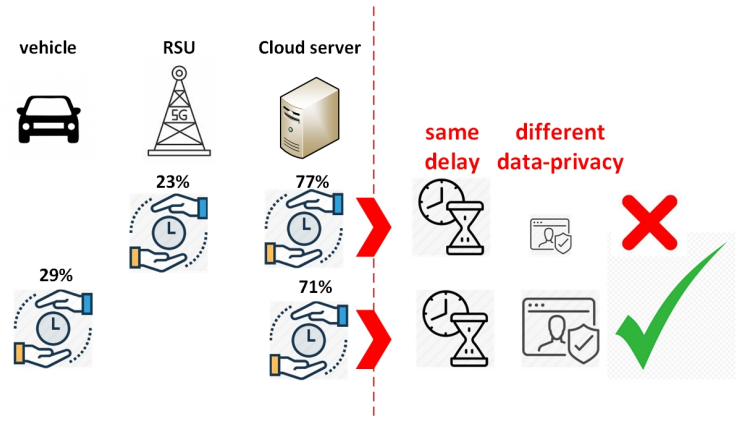


Figure 1. Preliminary analysis on end-to-end delay and data privacy.

Our preliminary analysis shows that this is a complex multiobjective problem since the two solutions having the same amount of end-to-end delay may have completely different data privacy. In [27], a multiobjective dynamic task allocation problem, as in our case, is formalized by integer linear optimization and is shown to have a nondetermin-

istic polynomial time (NP)-hard solution. As the processing constraints of 5G network devices are considered, it is easily concluded that such an NP-hard problem cannot be implementable by using exhaustive solutions. Optimization algorithms such as genetic algorithm attain approximating solutions for NP-hard search spaces [30]. It is emphasized in [31] that the genetic-algorithm-based optimization solutions have better performance characteristics meeting 5G requirements. A genetic-algorithm-based solution is inspired by the nature of evolution, including four main phases: parent selection, recombination, mutation, and survivor selection. It overpasses the exhaustive searching approach in terms of less processing time while finding the optimal solution. Therefore, there are many studies in the literature that propose genetic-algorithm-based improvements for machine learning models. For example, in [32], a genetic algorithm is used for online partitioning for edge intelligent applications and it results in less time than conventional solutions to obtain an optimal plan, which makes it implementable on edge devices. In [33], a genetic algorithm is executed to perform parameters' fine-tuning for machine learning on a cloud server. Thanks to its nature, it accelerates model training and enhances the precision result. In [34], ANN-based stock market predictions are improved by the genetic algorithm optimization process in order to meet the timing requirements of the system. In [35], the genetic algorithm is preferred to decrease training time by performing feature selection for machine learning algorithms; therefore, it can be executable in low-capacity edge devices. Here, thanks to the high retrainability of the genetic algorithm, the features required to train the model can be decreased and it can be executable in local devices.

1.3. Contributions

The related works and preliminary analysis show that the existing studies do not compare edge-AI-based learning and centralized learning approaches in terms of precision, end-to-end delay, and data privacy. While they address handling learning models on edge devices to maintain data privacy, they suffer from the loss of precision that could be achieved in a centralized cloud server. In comparison to other artificial intelligence (AI) techniques, such as neural networks or support vector machines, the genetic algorithm was specifically chosen in the literature for its ability to efficiently handle multiobjective optimization in distributed environments like federated learning. Other AI algorithms could offer improvements in precision, but they would require more processing power and time, which is a significant limitation in edge computing environments. Moreover, the evolutionary approach in the genetic algorithm allows for better adaptability and faster convergence when the training location (vehicle devices, RSUs, or cloud servers) needs to be optimized. Hence, we selected the genetic algorithm to solve our multiobjective problem, which determines the training location of the federated learning model (whether on a vehicle device, on an RSU, or on the cloud server) by considering end-to-end delay and data privacy. Therefore, we propose a novel genetic algorithm optimized federated learning (GAoFL) approach for the joint consideration of the corresponding problems (i) to accelerate the solution of well-predicted end-to-end delay and (ii) to preserve the data privacy by minimizing the probability of data leakage attack according to the problem formulation.

When the aforementioned motivations are taken into consideration, the main contributions of the study can be stated as given below:

- A novel model training delay (T) is mathematically defined by including the propagation delay for taking a trained model from center to end-user and the model training delay.
- A novel data privacy parameter (P) is mathematically defined by assigning different weights to training models in vehicles, RSUs, and cloud servers.
- A novel dataset is generated by a genetic algorithm with three labels of T, named nonacceptable, acceptable, and best according to 5G delay requirements.
- A novel GAoFL approach according to a novel metric, T (training delay in server) divided by *Fmeasure* (training accuracy) is devised.

The rest of the paper is organized as follows: Section 2 gives the details of proposed system architecture in terms of the data acquisition, the evaluation metrics, and the proposed GAoFL approach. Section 3 evaluates the performance of the proposed method by comparing the studies in the literature. Section 4 summarizes the paper by providing conclusions.

2. Proposed System Architecture

In this section, the proposed system architecture for the GAoFL approach is given in detail. The running principle of the proposed algorithm is illustrated in Figure 2. There are 100 vehicles, 15 RSUs, and 1 cloud server in the topology which are all connected in real time and delineate an urban area [36–38]. Initially, each end-user device obtains its own data and runs the federated learning tasks thanks to the implementation of the proposed GAoFL approach by using a Java-based environment. The proposed approach is based on 4 basic steps of the genetic algorithm, stated by the number blocks in the figure: 1—parent selection, 2—recombination, 3—mutation, and 4—survivor selection. While training data, there is a solution pool that considers the randomly generated solutions, and each solution determines one of the task running schedules that is given in Section 2.1.

In the proposed algorithm, each device calculates the fitness value by computing the end-to-end delay (T) over data privacy (P) and *Fmeasure* values obtained from the candidate solution and learning rates. The calculation details are given in Sections 2.2 and 2.3, respectively. In each learning round, devices find the best solution according to fitness value and share it with the other devices. In the next generation round, according to the best solution details, the RSU server is also included in the learning rounds. The same steps are repeated for all end-user devices and also for RSU servers, and the best solution is shared among them. According to the best solution detail, the cloud server can also be included in learning rounds, and, in that case, the same steps are repeated until the algorithm is converged according to fitness value. Here, including RSU and cloud servers in the training process is determined in accordance with the solution details produced by the genetic algorithm output. If the best solution determines the running tasks in the RSU and/or cloud servers, then in the next round, the RSU and/or cloud servers also run the genetic algorithm and share the best solution that they find.

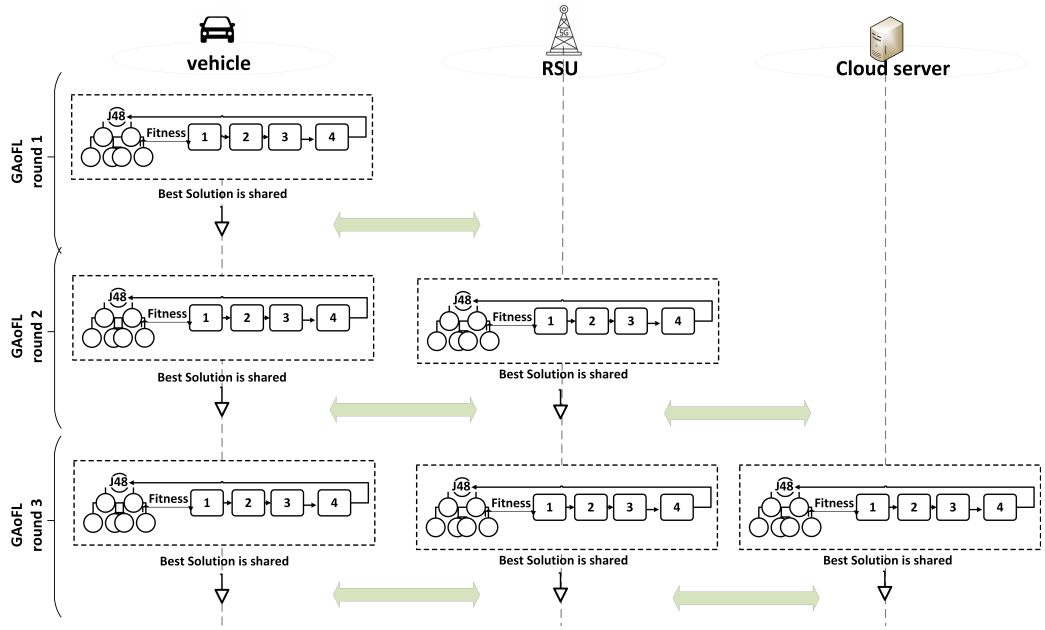


Figure 2. The proposed system architecture for the GAoFL approach based on genetic algorithm with 4 steps: 1—parent selection, 2—recombination, 3—mutation, and 4—survivor selection.

2.1. Data Acquisition

In Table 1, the features of the dataset that is generated by the proposed genetic algorithm are given. There are 101 features, where the first 100 of them are real numbers, and the last one (T) is class. The first 100 features, from S1 to S100, represent the identification number of servers, where they are located in the cloud, RSU, or vehicles in 5G vehicular networks. There is a cloud server for centralized learning, 15 RSUs, and 100 vehicles that can act as computing devices for the federated learning by the Java-based environment. If the data are between 0 and 99, it means that the federated learning is performed on the end-user device; whereas if the data are between 100 and 114 it means that the RSU is used for machine learning, and 115 means that the centralized learning is performed on the cloud server.

Table 1. The dataset generated based on the genetic algorithm.

S1 (REAL)	S2 (REAL)	...	S100 (REAL)	*P* (REAL)	*T* (CLASS LABEL)
0	0	...	0	1	*NON-ACCEPTABLE*
0	0	...	40	0.91	*NON-ACCEPTABLE*
100	10	...	85	0.85	*ACCEPTABLE*
40	115	...	10	0.793	*ACCEPTABLE*
100	10	...	115	0.549	*BEST*
115	70	...	115	0.192	*BEST*

T (in seconds) is the training duration of the models on the cloud, the RSU, or vehicles. It is calculated as follows [39,40]:

$$T = \frac{Training\ Data\ Size \cdot CPU}{Computing} + T_{Prop}(Model\ Source,\ End\ User) \quad (1)$$

where training data size varies between 1 KB and 1 GB, the CPU requires a CPU cycle per bit to train the model for the server, and it is taken as 100 bits per cycle, and computing is the capability of the server to train a model. If the federated learning is performed on a vehicle, computing capability is assumed as 0.5×10^9 bits/s in the calculation of T. If the federated learning is performed on an RSU, computing capability is assumed as 10×10^9 bits/s, and if the centralized learning is performed on the cloud, computing capability is assumed as 100×10^9 bits/s [41]. $T_{Prop}(target, user)$ defines the propagation delay to take a trained model from the RSU or cloud servers to the vehicle. If the target is a vehicle device, then the propagation is taken as 0. On the other hand, if the cloud server is used to train the model, it obviously has less training delay than the RSU and the vehicle due to having more computational capability. However, an additional propagation delay of 0.05 ms is considered to transfer the model from the cloud to the vehicle [39]. Similarly, the propagation time between the RSU and the vehicle is taken as 0.001 ms [40]. They can be ignored while considering the processing times on the vehicle, the RSU, and the cloud servers. According to two thresholds on model training delay, T has three classes, using Equation (2) as follows:

$$T_{LABEL} = \begin{cases} NON\text{-}ACCEPTABLE, & 4\ \text{ms} < T \\ ACCEPTABLE, & 1\ \text{ms} < T < 4\ \text{ms} \\ BEST, & T < 1\ \text{ms} \end{cases} \quad (2)$$

Moreover, to define the data privacy parameter (P) mathematically, we propose the following formula:

$$P = L_{vehicle} + 0.5 \cdot L_{RSU} + 0.1 \cdot L_{cloud} \quad (3)$$

where P is calculated by the weighted average of task loads in each device: vehicle, RSU, and cloud server. According to Table 1, if the data are between 0 and 99, it means that the federated learning is performed on a vehicle; $L_{vehicle}$ is calculated by the ratio of tasks that would be assigned to the vehicle. If the data are between 100 and 114, it means that an RSU is used for federated learning and L_{RSU} is calculated by the ratio of tasks that would be assigned to the RSU. If the data are 115, it means that the centralized learning is performed in the cloud server, and L_{cloud} is calculated by the ratio of tasks that would be assigned to the server as given below:

$$L_{server} = \begin{cases} \frac{\sum_i^N SL_i \cdot TaskSize_i \cdot (i \in vehicle)}{\sum_i^N S_i \cdot TaskSize_i}, & server\ is\ vehicle \\ \frac{\sum_i^N SL_i \cdot TaskSize_i \cdot (i \in RSU)}{\sum_i^N S_i \cdot TaskSize_i}, & server\ is\ RSU \\ \frac{\sum_i^N SL_i \cdot TaskSize_i \cdot (i \in cloud)}{\sum_i^N S_i \cdot TaskSize_i}, & server\ is\ cloud \end{cases} \tag{4}$$

where N is the number of tasks, which is taken as 100, and the server location, SL_i, is either 0 or 1, and determines the task assignment, as shown in Table 1. $TaskSize_i$ is the size of the task per vehicle, which is taken as 1 MB in our scenario. In Equation (3), the weight of the training task on the vehicle is taken as 1 because it fully preserves data privacy on the contrary executing tasks in RSU and cloud. On the other hand, the weight of the training task on the RSU server is selected as 0.5 because end-users prefer not to share their own data to the centralized cloud server. The L defines the loads of tasks in the devices and it changes between 0 and 1.

2.2. Evaluation Metrics

According to the features, the models are trained in vehicles, RSUs, and cloud servers independently. In machine learning literature, the performance of classifiers is usually measured with accuracy metrics. However, unbalanced datasets having uneven class distributions, as in our case, require the use of the *Fmeasure* metric, which can be calculated using precision and recall, defined in Equations (5) and (6), respectively. *Fmeasure* is used as the main performance metric for all the algorithms through this study and it is defined in Equation (7) [42]:

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

$$Fmeasure = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{7}$$

where TP, TN, FP, and FN are defined as true positive, true negative, false positive, and false negative predictions. In the representations, TP and TN are the predictions that are the same as the class labels through the test. On the other hand, FP and FN are false predictions and are different to the test sample labels [43].

2.3. The Proposed GAoFL Approach

In Algorithm 1, the proposed GAoFL approach is given. It takes the current statistics of vehicles, RSUs, and cloud servers from the physical layer. It returns the dataset and trained model that are aggregated by each generation of the genetic algorithm. In line 1, the algorithm initializes vehicles, RSUs, and cloud servers according to the taken statistics. In line 2, the solution pool with 100 chromosomes is initialized randomly. An example chromosome pool can be found in Table 2. The chromosomes are stored in an array list throughout the execution of the genetic-algorithm-based GAoFL. The entries in the table are feasible solutions obtained by training capacities of vehicles, RSUs, and the cloud server. If the end-to-end delay (T) could not take the BEST label in the dataset, the *Fmeasure* would

not be meaningful for the BEST label. Therefore, the fitness value of this solution cannot be evaluated, as seen in C1 and C2. In Algorithm 1, between lines 3 and 18, the federated learning rounds are executed, and in each of them there are 1000 iterations to create a new generation pool of chromosomes to find the best solution that would have the lowest end-to-end delay (T), higher data privacy (P), and higher $Fmeasure$. While selecting the optimal machine learning algorithm, we compared J48 [44,45], support vector machine (SVM) [46], decision table [47], and random forest [48]. Finally, we selected J48 to have the lowest training delay near the best $Fmeasure$. Each round should be finalized under 1 s to satisfy the 5G delay requirements. The corresponding analysis is detailed in Section 3.

Algorithm 1 Genetic algorithm optimized federated learning.

Require: Take statistics from physical topology
Ensure: Dataset and Trained model
1: Initialize vehicles, RSUs, and Cloud server to train data
2: Initialize solution pool with the feasible chromosomes by checking total training capacities of end-servers
3: **for each** learning round **do**
4: **for each** iteration **do**
5: Calculate end-to-end delay(T) by Equation (1)
6: Train data by using J48 learning algorithm and calculate Fmeasure
7: Calculate *Fitness* value $T/(P \cdot Fmeasure)$ by Equation (8)
8: Parent Selection according to $T/(P \cdot Fmeasure)$
9: Recombination by two-point cross-over
10: Mutation in one point
11: Update $T/(P \cdot Fmeasure)$ for each generation
12: Add solution pool to each datasets
13: Survivor Selection according to $T/(P \cdot Fmeasure)$
14: **end for**
15: **if** *Fitness* < 1 **then**
16: Stop learning
17: **end if**
18: **end for**
19: **return** Dataset and trained model

Table 2. An example of chromosome pool in GAoFL with calculated values.

Chromosome ID	Genes of Chromosome				Calculated Values of Chromosome			
	S1	**S2**	...	**S100**	***T***	***P***	***Fmeasure***	***Fitness***
C1	0	0	...	0	7.99	1	NAN	NAN
C2	0	103	...	115	1.94	0.846	NAN	NAN
C3	0	9	...	115	0.648	0.91	0.487	1.462
...	...	...	...	...	...	...	...	...
C100	50	101	...	114	0.41	0.555	1	0.738

In particular, J48 is known as the Java implementation of the J4.5 decision tree algorithm. The selection criteria of J48 in the proposed GAoFL algorithm can be explained as follows: The algorithm is favored for its high simplicity and rapid results, making it suitable for real-time applications [49]. The divide-and-conquer strategy in J48 efficiently partitions the data based on attributes with the highest information gain, which results in facilitating quick decision making. In 5G, where low latency is essential, J48's fast training process significantly enhances the responsiveness of vehicular communication. Furthermore, J48 excels in handling multicolumn datasets typical in 5G environments as a result of efficiently identifying relevant features and reducing complexity. Therefore, it can effectively classify the data and enable timely responses to changing conditions, especially in dynamic scenarios [50]. Integrating J48 into the fitness function optimizes model performance across

several generations, enhancing classification accuracy while minimizing training time. Existing studies in the literature show that decision trees in vehicular networks provide low latency and high data security. Thus, J48's rapid adaptability and effectiveness reinforce the GAoFL algorithm's success in real-time 5G vehicular networks.

Afterwards, to find best solution, the proposed genetic algorithm executes the following four phases which are listed below:

- Parent Selection: After calculating each fitness value as T by using Equation (1), each chromosome has an estimated data train delay. Moreover, the P is calculated by using Equation (3), and $Fmeasure$ is calculated by using Equation (7) on the trained data. Accordingly, the fitness function is calculated as given below:

$$Fitness = \frac{T}{P \cdot Fmeasure} \tag{8}$$

  According to $Fitness$ values, each chromosome is sorted from low to high, and the best 10 chromosomes that have less T, higher P, and higher $Fmeasure$ are selected as parents of the new generation pool. The learning rounds are stopped when $Fmeasure$ is less than 1.
- Recombination: Each parent is labeled as mother and father chromosomes randomly. They are also randomly paired. From the first gene to the randomly defined gene, the new generation takes the mother's genes. The remained ones are taken from the father's gene. After the combination, the new generation can be unfeasible. Therefore, by checking the total training capacity of vehicles, RSUs, and cloud servers, the solution is taken to feasible space.
- Mutation: Each new generation mutates over a randomly determined gene. This is performed to increase the possibility of having a better generation than the parents. Without this phase, the generation can repeat itself and the algorithm can become stuck on a single point. After this step, the new generation pool is added into the dataset.
- Survivor Selection: After new generation is created, the better ones according to $Fitness$ are selected for the next iteration. The remaining 10 chromosomes are deleted.

3. Performance Evaluation

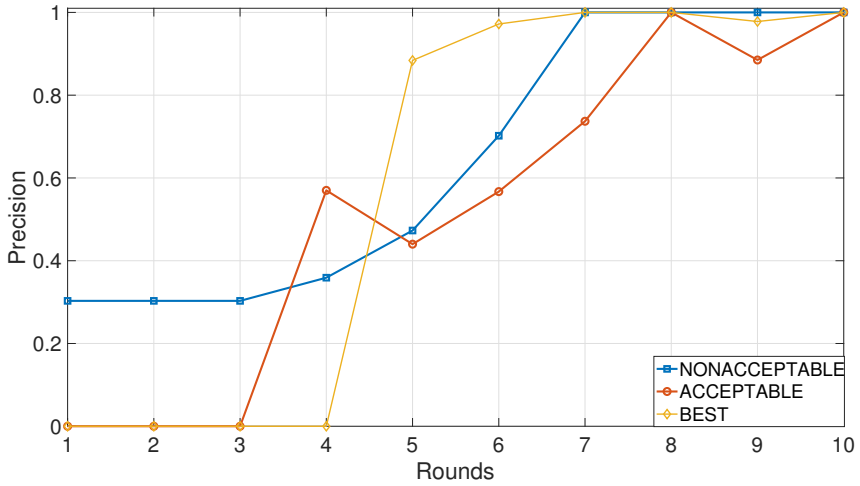
The performance of the proposed algorithm is evaluated by a Java-based environment and Weka 3.6.12 by using a MAC OSx Intel i5 processor and 8 GB RAM. The proposed GAoFL algorithm is compared by centralized, adaptive federated, and pure federated learning algorithms [25]. The evaluations are repeated with respect to different task sizes, where N is 100, 200, and 400. Moreover, J48, SVM, random forest, and decision table are compared according to the effect on the processing time of the proposed algorithm with respect to increasing iteration number and task sizes.

When N is 100, the number of solutions for trained data in each generation of the algorithm is given in Table 3. Here, in 10 rounds, the proposed algorithm converges in terms of fitness value ($T/(P{\cdot}Fmeasure)$). In the initial case, 16,807 solutions are obtained as nonacceptable according to the end-to-end delay (T) equation defined in Equation (1). In the second round, the proposed GAoFL approach in end-user devices results in better T values owing to the addition of a solution pool of 5 nonacceptable, 37 acceptable, and 1 best solutions. In the third round, vehicles and the RSUs also generate their solution pool, and the total solutions in the topology are achieved as 16,815 nonacceptable, 51 acceptable, and 4 best. The cloud server provides a solution pool after the 5th round of the proposed algorithm. In each round, the solutions of acceptable and best increase, but the nonacceptable solutions might decrease. This is caused by electing some solutions labeled as nonacceptable after the survival selection phase of the genetic algorithm. Though we are using $Fmeasure$ as a performance metric throughout the study, we also want to observe the changes in the constituents of $Fmeasure$, i.e., precision and recall. In more precise terms, for a classification problem, there is a trade-off between precision and recall. An acceptable prediction performance in terms of precision–recall may vary depending on the problem

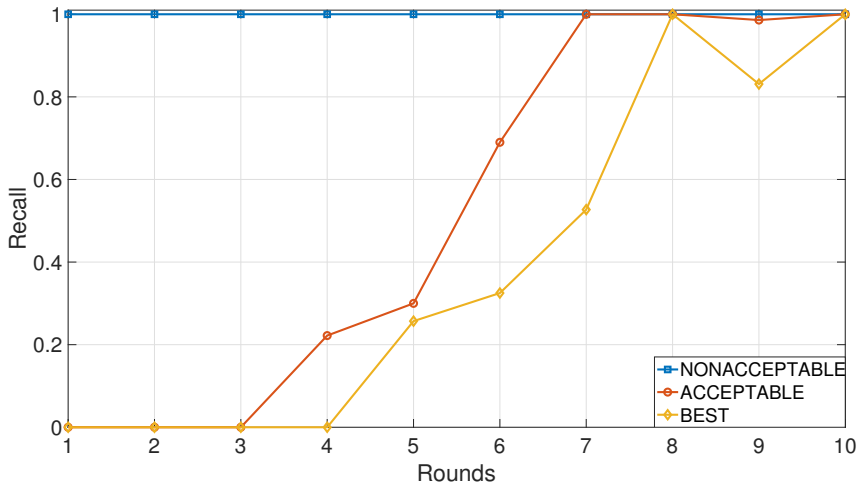
or the class to be predicted. In our case, precision–recall both are roughly maximized, i.e., become 1, after roughly round 8, and the results are given in Figure 3.

Table 3. The number of solutions for trained data in each genetic algorithm round when N is 100.

Round	Nonacceptable	Acceptable	Best	Data Location
1st	16,807	0	0	Vehicle
2nd	16,812	37	1	Vehicle
3rd	16,815	51	4	Vehicle, RSU
4th	16,810	61	3	Vehicle, RSU
5th	16,817	66	3	Vehicle, RSU, and cloud servers
6th	16,819	83	5	Vehicle, RSU, and cloud servers
7th	16,829	176	9	Vehicle, RSU, and cloud servers
8th	16,854	354	20	Vehicle, RSU, and cloud servers
9th	16,866	409	30	Vehicle, RSU, and cloud servers
10th	16,931	590	34	Vehicle, RSU, and cloud servers



(**a**) Precision results



(**b**) Recall results

Figure 3. Precision and recall results of the proposed GAoFL approach for each round when N is 100.

In Figure 4, *Fmeasure* and end-to-end delay (*T*) outcomes of the proposed algorithm for each round are given. In the left y-axis, *Fmeasure* is given by the bar graph, whereas

in the right y-axis, end-to-end delay (T) is given by the line graph. According to the result, as the rounds increase, the *Fmeasure* for the average of each label (nonacceptable, acceptable, best) increases gradually. It reaches 1 when the algorithm is in the eighth round. There is a slight decrease in the ninth round, but the algorithm converges in the tenth round with the *Fmeasure* as 1. On the other hand, the proposed algorithm finds the best task training solutions from the second round to the tenth. Namely, except the initial round where the training is only performed on end-user devices with pure federated learning, the proposed algorithm can give best training scheduling in terms of end-to-end delay (T) to the topology. Moreover, the prediction performance in terms of *Fmeasure* reaches 1 under 10 rounds as the round increases.

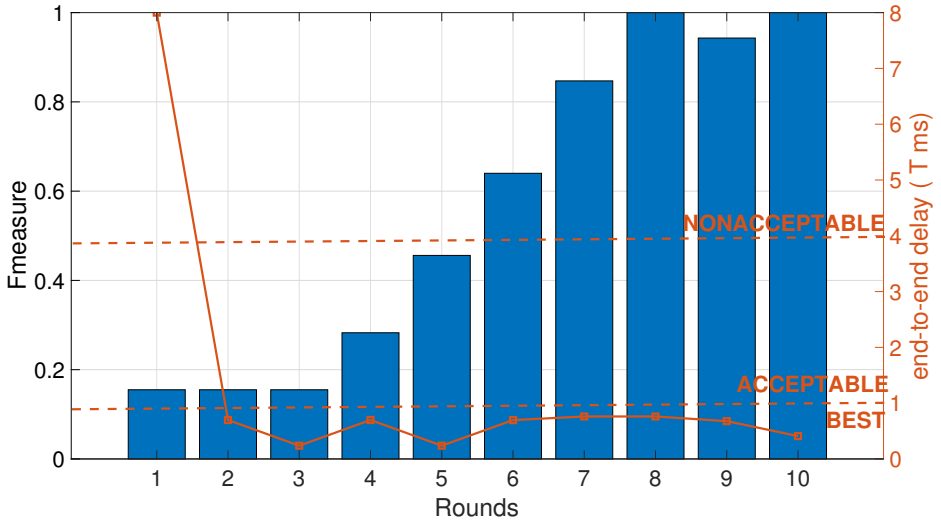


Figure 4. *Fmeasure* and T results of proposed GAoFL approach for each round when N is 100.

Tables 4–6 give the overall results of the proposed GAoFL approach for task distribution among computational locations in vehicular networks when the number of tasks (N) is increased to 100, 200, and 400. All of end-to-end delay (T), data privacy (P), *Fmeasure* and fitness values are jointly analyzed in these tables. As shown by the results in Table 4, there are different best solutions for training tasks as the rounds increase. In particular, the *Fmeasure* can be calculated after the fifth round for the label BEST, although the average of *Fmeasure* is nearly 0.5, as seen in Figure 4. *Fmeasure* for BEST label also reaches 1 after the eighth round, and the proposed algorithm converges in the tenth round. Here, predicting the performance of the BEST label is more vital than the other labels because it means keeping the end-user quality of service under 1 ms, according to 5G key performance indicators. The proposed algorithm also considers the data privacy parameter (P) while choosing the example solution. Between the fourth and the seventh rounds, the fitness value of the algorithm fluctuates despite it outputting the same delay and privacy. The GAoFL leads to a penalty on fitness in each iteration to not become stuck in local optimum values. In fact, in the seventh round, the time and privacy parameters take nearly the best values, but *Fmeasure* for BEST is not at around 1. On the other hand, the *Fmeasure* value convergences to 1 in the tenth round of the algorithm, where 11% of the task training is assigned to the vehicle, 89% of the task training is assigned to the RSU servers, and none of the task training is assigned to the cloud server. It finally produces a 0.41 ms end-to-end delay (T) which is under BEST threshold and a 0.555 data privacy parameter (P) when *Fmeasure* is 1, which results in the minimum fitness value of 0.738. In Tables 5 and 6, the algorithm reaches the minimum fitness value in the 11th and 15th rounds when the task numbers are 200 and 400, respectively. As the task numbers increase, running the task locally results in unacceptable time because of the increased load on each server. Therefore, finding a solution with the BEST label becomes harder in each round and iteration. When N is 200, the algorithm stops with 0.41 ms end-to-end delay (T) and 0.655 data privacy parameter (P) when the *Fmeasure*

is 1, which results in the minimum fitness value of 0.74. When N is 400, the algorithm runs 15 rounds to find the minimum fitness value, and it results in a 0.13 ms end-to-end delay (T) and a 0.157 data privacy parameter (P) when *Fmeasure* is 1. Due to the extreme task load, the tasks can be mostly executed on the cloud server, resulting in low data privacy. To handle this, the capacity of RSUs or vehicles should be increased.

Table 4. Example BEST solutions for different rounds in the proposed algorithm when N is 100.

Round	Example Solution for Training			T (ms)	P [0–1]	*Fmeasure* for BEST	*Fitness*
	$L_{vehicle}$	L_{RSU}	L_{cloud}				
1	1	0	0	7.99	1	NAN	NAN
2	0.29	0	0.71	0.698	0.37	NAN	NAN
3	0	0.77	0.23	0.237	0.408	NAN	NAN
4	0	0.23	0.77	0.677	0.192	NAN	NAN
5	0	0.77	0.23	0.237	0.408	0.398	1.459
6	0.29	0.23	0.48	0.69	0.453	0.487	3.127
7	0.29	0.47	0.24	0.761	0.549	0.847	1.636
8	0.29	0.47	0.24	0.761	0.549	1	1.386
9	0.29	0	0.71	0.677	0.361	0.943	1.98
10	0.11	0.89	0	0.41	0.555	1	0.738

Table 5. Example BEST solutions for different rounds in the proposed algorithm when N is 200.

Round	Example Solution for Training			T (ms)	P [0–1]	*Fmeasure* for BEST	*Fitness*
	$L_{vehicle}$	L_{RSU}	L_{cloud}				
1	1	0	0	16	1	NAN	NAN
2	0.11	0.88	0	0.83	0.5575	NAN	NAN
3	0.06	0.83	0.105	0.61	0.488	NAN	NAN
4	0.065	0.83	0.105	0.63	0.49	NAN	NAN
5	0.05	0.845	0.105	0.59	0.483	0.23	5.311
6	0.115	0.885	0	0.7	0.5575	0.398	3.154
7	0.04	0.795	0.165	0.53	0.454	0.487	2.397
8	0.05	0.515	0.435	0.37	0.351	0.645	1.634
9	0.04	0.795	0.165	0.53	0.454	1	1.167
10	0.045	0.765	0.19	0.501	0.4465	1	1.122
11	0.55	0.15	0.3	0.485	0.655	1	0.74

In Figure 5, the overall comparison of the proposed GAoFL approach and the conventional learning approaches, including centralized learning, adaptive federated learning, and pure federated learning, are given. These results were obtained when N is 100. The left y-axis shows the end-to-end delay (T) calculated by Equation (1), whereas the right y-axis shows the data privacy parameter (P) calculated by Equation (3). The red line plot shows the end-to-end delay, whereas the green line plot shows the data privacy parameter according to four different learning approaches. The bar graph represents the training task details for each of the learning approach results. In centralized learning, there is the deep learning approach, which aggregates data from the whole topology without considering the data privacy parameter and the training model in the centralized cloud server. Therefore, it is the best solution ever to minimize end-to-end delay (T); however, it suffers from data

privacy since P is around 0.1, a relatively low value. In the proposed GAoFL approach, the predicted solution results in end-to-end delay under the BEST label. It trains 11%, 89%, and 0% of the tasks in vehicles, RSUs, and cloud servers, respectively. Owing to considering both delay and data privacy, the P value reaches nearly 0.55, which is quite higher than the one obtained in the centralized learning approach. The other federated learning approaches suffer in terms of end-to-end delay performance. They cannot predict the best solution. They can advise only nonacceptable solutions. Therefore, providing a data privacy parameter around 0.8 and 1 does not make sense against the end-to-end delay damages. Here, adaptive learning proposes to train 77% and 23% of the tasks in end-user devices and the cloud server, respectively. In the pure federated learning, only end-user devices are used for the task training, which results in the worst end-to-end delay performance.

Table 6. Example BEST solutions for different rounds in the proposed algorithm when N is 400.

Round	Example Solution for Training			T (ms)	P [0–1]	*Fmeasure* for BEST	*Fitness*
	$L_{vehicle}$	L_{RSU}	L_{cloud}				
1	1	0	0	32	1	NAN	NAN
2	0.8	0.16	0.037	20.6	0.886	NAN	NAN
3	0.715	0.155	0.13	16.3	0.8055	NAN	NAN
4	0.7025	0.157	0.14	15	0.795	NAN	NAN
5	0.4	0.095	0.5	5.2	0.5	NAN	NAN
6	0.3	0.645	0.052	3.59	0.63	NAN	NAN
7	0.29	0.665	0.045	3.39	0.627	NAN	NAN
8	0.295	0.69	0.015	3.5	0.641	0.48	11.36
9	0.28	0.695	0.025	3.2	0.63	0.8	6.349
10	0.195	0.047	0.75	1.2	0.294	1	4.074
11	0.17	0.057	0.765	1.03	0.282	1	3.642
12	0.17	0.04	0.79	0.78	0.274	1	3.573
13	0.11	0.035	0.855	0.41	0.213	1	1.924
14	0.072	0.027	0.9	0.2	0.176	1	1.134
15	0.055	0.02	0.925	0.13	0.157	1	0.825

In Figure 6, different machine learning algorithms used in the proposed GAoFL approach are analyzed in terms of the processing time of GAoFL. The number of tasks (N) is increased from 100 to 400; namely, the total data task size to train the model is increased from 16,000 to 67,000 when there are 100 vehicles in the topology. The processing time should be under 1 s. If it exceeds 1 s, this means that the system capacity is full, and a capacity increase is required on the servers. In each round, vehicles, RSUs, and cloud servers should terminate the training process within 1 s. Therefore, each iteration per device should be finished under this determined time interval of the rounds. We compared J48, SVM, random forest, and decision table [44–48] in terms of processing time with different iteration numbers of the genetic algorithm where there are 100 chromosomes in the solution pool. In the figure, y-axes show the processing time, whereas x-axes show the increased iteration number of genetic algorithm in each round. The bar labels show the different task training sizes in the topology. Different machine learning algorithms result in the same *Fmeasure* for different training tasks. All machine learning algorithms can give a response under 1 s when the iteration number of the genetic algorithm is 10 and the number of vehicles is 100. However, as the number of tasks increases, random forest and decision table cannot satisfy the response time requirements. On the other hand, in order to enhance

genetic algorithm performance to find the optimal solution that minimizes end-to-end delay by preserving data privacy, the iteration number should be 1000. Therefore, only J48 is capable of responding under the timing requirements in each round. As a result, J48 and SVM are usable when the number of tasks and iteration number are low, whereas J48 is the only choice in the proposed GAoFL approach when the number of tasks and iteration number are high.

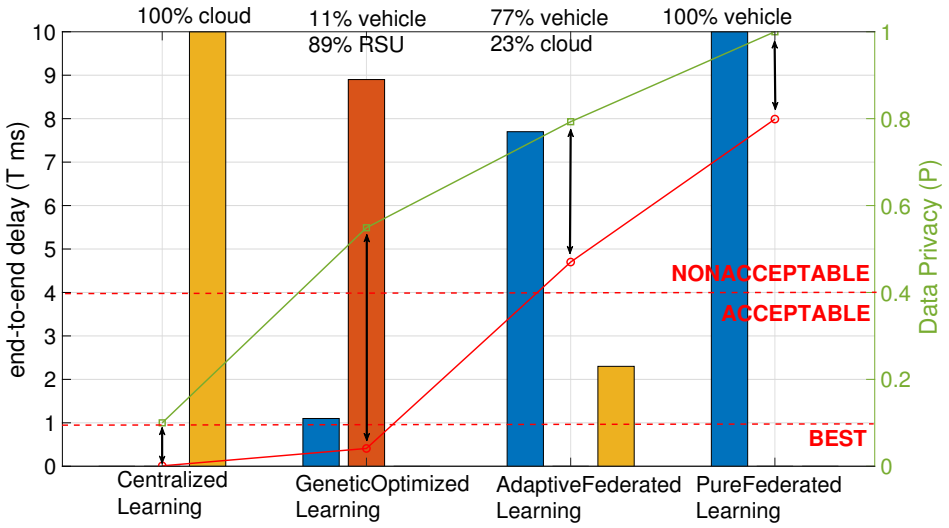


Figure 5. Overall comparison of proposed GAoFL approach and the other approaches in the literature.

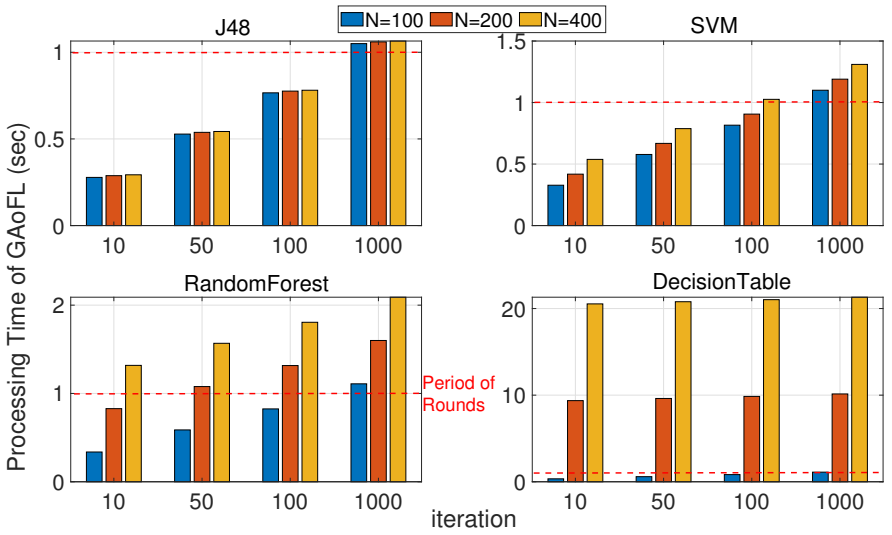


Figure 6. The processing time analysis of GAoFL with different machine learning algorithms in terms of increased iteration number and tasks.

4. Conclusions

In this paper, we proposed a GAoFL approach that jointly considers minimizing end-to-end delay and preserving data privacy for vehicular computing. The proposed algorithm determines the location of model training among vehicles, RSUs, and the cloud server, respectively. To evaluate the performance of the proposed method computationally, a novel dataset based on a genetic algorithm was generated where the end-to-end delay (T) is labeled as nonacceptable, acceptable, and best. First, in the proposed method, the end-to-end delay and data privacy are analytically defined. Then, depending on the newly defined fitness function (end-to-end delay/data privacy·*Fmeasure*), the proposed genetic algorithm finds the optimal solution that determines model training locations by executing

four evolutionary phases: parent selection, recombination, mutation, and survivor selection. Accordingly, the proposed GAoFL approach is evaluated on the generated data to solve delay prediction and data privacy optimization problems. Moreover, it is also compared with the centralized, pure-federated, and adaptive-federated learning approaches from the literature of vehicular computing. The experimental results have shown that our algorithm is better than the algorithms proposed in the literature in terms of high delay prediction performance while preserving data privacy. As a future work, we will focus on model training and the joint optimization according to the different requirements of 5G contents in vehicular networks, including eMBB, URLLC, and mIoT.

Author Contributions: Conceptualization, M.E.-Ö. and F.Y.; methodology, A.Ö. and Y.Ö.; software, Y.Ö. and F.Y.; validation, M.E.-Ö., A.Ö. and F.Y.; formal analysis, M.E.-Ö. and A.Ö.; investigation, Y.Ö. and F.Y.; resources, M.E.-Ö., Y.Ö. and F.Y.; data curation, M.E.-Ö. and A.Ö.; writing—original draft preparation, M.E.-Ö., A.Ö., Y.Ö. and F.Y.; writing—review and editing, M.E.-Ö., A.Ö., Y.Ö. and F.Y.; visualization, M.E.-Ö. and Y.Ö.; supervision, A.Ö.; project administration, F.Y. All authors have read and agreed to the published version of the manuscript.

Funding: Müge Erel-Özçevik is supported by The Scientific and Technological Research Council of Turkey (TUBITAK) 1515 Frontier R&D Laboratories Support Program for BTS Advanced AI Hub: BTS Autonomous Networks and Data Innovation Lab. Project 5239903.

Data Availability Statement: The data and source codes will be made available on reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Kong, Q.; Su, L.; Ma, M. Achieving Privacy-Preserving and Verifiable Data Sharing in Vehicular Fog with Blockchain. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 4889–4898. [CrossRef]
2. Elayoubi, S.E.; Jemaa, S.B.; Altman, Z.; Galindo-Serrano, A. 5G RAN Slicing for Verticals: Enablers and Challenges. *IEEE Commun. Mag.* **2019**, *57*, 28–34. [CrossRef]
3. *Ericsson Mobility Report*; Tech. Rep. EAB-22:010742 Uen Rev D; Technical Report; Ericsson: Stockholm, Sweden, 2022.
4. Orabi, M.; Al Aghbari, Z.; Kamel, I. FogLBS: Utilizing fog computing for providing mobile Location-Based Services to mobile customers. *Pervasive Mob. Comput.* **2023**, *94*, 101832. [CrossRef]
5. Tang, W.; Gao, D.; Yu, S.; Lu, J.; Wei, Z.; Li, Z.; Chen, N. Reliable and adaptive computation offload strategy with load and cost coordination for edge computing. *Pervasive Mob. Comput.* **2024**, *102*, 101932. [CrossRef]
6. Moshawrab, M.; Adda, M.; Bouzouane, A.; Ibrahim, H.; Raad, A. Securing Federated Learning: Approaches, Mechanisms and Opportunities. *Electronics* **2024**, *13*, 3675. [CrossRef]
7. Wang, S.; He, D.; Yang, M.; Duo, L. Cost-aware task offloading in vehicular edge computing: A Stackelberg game approach. *Veh. Commun.* **2024**, *49*, 100807. [CrossRef]
8. Zhou, Z.; Luo, X.; Wang, Y.; Mao, J.; Luo, F.; Bai, Y.; Wang, X.; Liu, G.; Wang, J.; Zeng, F. A Practical Data Audit Scheme with Retrievability and Indistinguishable Privacy-Preserving for Vehicular Cloud Computing. *IEEE Trans. Veh. Technol.* **2023**, *72*, 16592–16606. [CrossRef]
9. Lohrasbinasab, I.; Shahraki, A.; Taherkordi, A.; Delia Jurcut, A. From statistical- to machine learning-based network traffic prediction. *Trans. Emerg. Telecommun. Technol.* **2022**, *33*, e4394. [CrossRef]
10. Alsayfi, M.S.; Dahab, M.Y.; Eassa, F.E.; Salama, R.; Haridi, S.; Al-Ghamdi, A.S. Securing Real-Time Video Surveillance Data in Vehicular Cloud Computing: A Survey. *IEEE Access* **2022**, *10*, 51525–51547. [CrossRef]
11. Boutaba, R.; Salahuddin, M.A.; Limam, N.; Ayoubi, S.; Shahriar, N.; Estrada-Solano, F.; Caicedo, O.M. A comprehensive survey on machine learning for networking: Evolution, applications and research opportunities. *J. Internet Serv. Appl.* **2018**, *9*, 1–99. [CrossRef]
12. Xia, X.; Chen, F.; He, Q.; Cui, G.; Grundy, J.; Abdelrazek, M.; Bouguettaya, A.; Jin, H. OL-MEDC: An Online Approach for Cost-Effective Data Caching in Mobile Edge Computing Systems. *IEEE Trans. Mob. Comput.* **2023**, *22*, 1646–1658. [CrossRef]
13. Gao, B.; Zhou, Z.; Liu, F.; Xu, F.; Li, B. An Online Framework for Joint Network Selection and Service Placement in Mobile Edge Computing. *IEEE Trans. Mob. Comput.* **2022**, *21*, 3836–3851. [CrossRef]
14. Wang, F.; Zhang, M.; Wang, X.; Ma, X.; Liu, J. Deep Learning for Edge Computing Applications: A State-of-the-Art Survey. *IEEE Access* **2020**, *8*, 58322–58336. [CrossRef]
15. Rodrigues, T.K.; Suto, K.; Nishiyama, H.; Liu, J.; Kato, N. Machine Learning Meets Computation and Communication Control in Evolving Edge and Cloud: Challenges and Future Perspective. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 38–67. [CrossRef]
16. Zheng, C.; Liu, S.; Huang, Y.; Zhang, W.; Yang, L. Unsupervised Recurrent Federated Learning for Edge Popularity Prediction in Privacy-Preserving Mobile-Edge Computing Networks. *IEEE Internet Things J.* **2022**, *9*, 24328–24345. [CrossRef]

17. McMahan, H.B.; Moore, E.; Ramage, D.; y Arcas, B.A. Federated Learning of Deep Networks using Model Averaging. *arXiv* **2016**, arXiv:1602.05629.
18. Wu, M.; Cheng, G.; Ye, D.; Kang, J.; Yu, R.; Wu, Y.; Pan, M. Federated Split Learning with Data and Label Privacy Preservation in Vehicular Networks. *IEEE Trans. Veh. Technol.* **2024**, *73*, 1223–1238. [CrossRef]
19. Wu, N.; Lin, X.; Lu, J.; Zhang, F.; Chen, W.; Tang, J.; Xiao, J. Byzantine-Robust Multimodal Federated Learning Framework for Intelligent Connected Vehicle. *Electronics* **2024**, *13*, 3635. [CrossRef]
20. Yue, K.; Jin, R.; Wong, C.W.; Baron, D.; Dai, H. Gradient obfuscation gives a false sense of security in federated learning. In Proceedings of the 32nd USENIX Conference on Security Symposium, Anaheim, CA, USA, 9–11 August 2023; SEC '23.
21. Yang, Y.; Hui, B.; Yuan, H.; Gong, N.; Cao, Y. PrivateFL: Accurate, Differentially Private Federated Learning via Personalized Data Transformation. In Proceedings of the 32nd USENIX Security Symposium (USENIX Security 23), Anaheim, CA, USA, 9–11 August 2023; pp. 1595–1612.
22. Li, Y.; Tao, X.; Zhang, X.; Liu, J.; Xu, J. Privacy-Preserved Federated Learning for Autonomous Driving. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 8423–8434. [CrossRef]
23. Batool, H.; Anjum, A.; Khan, A.; Izzo, S.; Mazzocca, C.; Jeon, G. A secure and privacy preserved infrastructure for VANETs based on federated learning with local differential privacy. *Inf. Sci.* **2024**, *652*, 119717. [CrossRef]
24. Wang, S.; Tuor, T.; Salonidis, T.; Leung, K.K.; Makaya, C.; He, T.; Chan, K. Adaptive Federated Learning in Resource Constrained Edge Computing Systems. *IEEE J. Sel. Areas Commun.* **2019**, *37*, 1205–1221. [CrossRef]
25. Tyou, I.; Murata, T.; Fukami, T.; Takezawa, Y.; Niwa, K. A Localized Primal-Dual Method for Centralized/Decentralized Federated Learning Robust to Data Heterogeneity. *IEEE Trans. Signal Inf. Process. Netw.* **2024**, *10*, 94–107. [CrossRef]
26. Guendouzi, B.S.; Ouchani, S.; EL Assaad, H.; EL Zaher, M. A systematic review of federated learning: Challenges, aggregation methods, and development tools. *J. Netw. Comput. Appl.* **2023**, *220*, 103714. [CrossRef]
27. Mohammad, U.; Sorour, S.; Hefeida, M. Dynamic Task Allocation for Mobile Edge Learning. *IEEE Trans. Mob. Comput.* **2023**, *22*, 6860–6873. [CrossRef]
28. Barbieri, L.; Savazzi, S.; Brambilla, M.; Nicoli, M. Decentralized federated learning for extended sensing in 6G connected vehicles. *Veh. Commun.* **2022**, *33*, 100396. [CrossRef]
29. Mahbub, M.; Shubair, R.M. Contemporary advances in multi-access edge computing: A survey of fundamentals, architecture, technologies, deployment cases, security, challenges, and directions. *J. Netw. Comput. Appl.* **2023**, *219*, 103726. [CrossRef]
30. Kouhalvandi, L.; Shayea, I.; Ozoguz, S.; Mohamad, H. Overview of evolutionary algorithms and neural networks for modern mobile communication. *Trans. Emerg. Telecommun. Technol.* **2022**, *33*, e4579. [CrossRef]
31. Majeed, O.K.; Hashim Ali, R.; Ijaz, A.Z.; Ali, N.; Arshad, U.; Imad, M.; Nabi, S.; Tahir, J.; Saleem, M. Performance comparison of genetic algorithms with traditional search techniques on the N-Queen Problem. In Proceedings of the 2023 International Conference on IT and Industrial Technologies (ICIT), Orlando, FL, USA, 4–6 April 2023; pp. 1–6. [CrossRef]
32. Na, J.; Zhang, H.; Lian, J.; Zhang, B. Genetic Algorithm-Based Online-Partitioning BranchyNet for Accelerating Edge Inference. *Sensors* **2023**, *23*, 1500. [CrossRef]
33. Sulaiman, M.; Farmanbar, M.; Belbachir, A.N.; Rong, C. Genetic Algorithm Empowering Unsupervised Learning for Optimizing Building Segmentation from Light Detection and Ranging Point Clouds. *Remote Sens.* **2024**, *16*, 3603. [CrossRef]
34. Chang, Y.H.; Huang, C.W. Utilizing Genetic Algorithms in Conjunction with ANN-Based Stock Valuation Models to Enhance the Optimization of Stock Investment Decisions. *AI* **2024**, *5*, 1011–1029. [CrossRef]
35. Silva, J.L.; Fernandes, R.; Lopes, N. Performance Study on the Use of Genetic Algorithm for Reducing Feature Dimensionality in an Embedded Intrusion Detection System. *Systems* **2024**, *12*, 243. [CrossRef]
36. Cheng, H.; Fei, X.; Boukerche, A.; Almulla, M. GeoCover: An efficient sparse coverage protocol for RSU deployment over urban VANETs. *Ad Hoc Netw.* **2015**, *24*, 85–102. [CrossRef]
37. Choi, C.S.; Baccelli, F. LOS Coverage Area in Vehicular Networks with Cox-Distributed Roadside Units and Relays. *IEEE Trans. Veh. Technol.* **2023**, *72*, 7772–7782. [CrossRef]
38. Feng, M.; Yao, H.; Ungurean, I. A Roadside Unit Deployment Optimization Algorithm for Vehicles Serving as Obstacles. *Mathematics* **2022**, *10*, 3282. [CrossRef]
39. Kiran, N.; Pan, C.; Wang, S.; Yin, C. Joint resource allocation and computation offloading in mobile edge computing for SDN based wireless networks. *J. Commun. Netw.* **2020**, *22*, 1–11. [CrossRef]
40. Bozkaya, E.; Erel-Özçevik, M.; Bilen, T.; Özçevik, Y. Proof of Evaluation-based energy and delay aware computation offloading for Digital Twin Edge Network. *Ad Hoc Netw.* **2023**, *149*, 103254. [CrossRef]
41. Bozkaya, E.; Canberk, B.; Schmid, S. Digital Twin-Empowered Resource Allocation for 6G-Enabled Massive IoT. In Proceedings of the Workshop on the Evolution of Digital Twin Paradigm in Wireless Communications, IEEE International Conference on Communications (ICC 2023), Rome, Italy, 28 May–1 June 2023; pp. 918–923.
42. Ahsan, R.; Shi, W.; Corriveau, J.P. Network intrusion detection using machine learning approaches: Addressing data imbalance. *IET Cyber-Phys. Syst. Theory Appl.* **2022**, *7*, 30–39. [CrossRef]
43. Binkhonain, M.; Zhao, L. A review of machine learning algorithms for identification and classification of non-functional requirements. *Expert Syst. Appl. X* **2019**, *1*, 100001. [CrossRef]
44. Yucalar, F. Developing an Advanced Software Requirements Classification Model Using BERT: An Empirical Evaluation Study on Newly Generated Turkish Data. *Appl. Sci.* **2023**, *13*, 11127. [CrossRef]

45. Wang, Z.; Gai, K. Decision Tree-Based Federated Learning: A Survey. *Blockchains* **2024**, *2*, 40–60. [CrossRef]
46. Ji, Y.; Lee, H. Event-Based Anomaly Detection Using a One-Class SVM for a Hybrid Electric Vehicle. *IEEE Trans. Veh. Technol.* **2022**, *71*, 6032–6043. [CrossRef]
47. Heiyanthuduwage, S.R.; Altas, I.; Bewong, M.; Islam, M.Z.; Deho, O.B. Decision Trees in Federated Learning: Current State and Future Opportunities. *IEEE Access* **2024**, *12*, 127943–127965. [CrossRef]
48. Xiang, P.; Zhou, L.; Tang, L. Transfer learning via random forests: A one-shot federated approach. *Comput. Stat. Data Anal.* **2024**, *197*, 107975. [CrossRef]
49. Brunello, A.; Marzano, E.; Montanari, A.; Sciavicco, G. J48SS: A Novel Decision Tree Approach for the Handling of Sequential and Time Series Data. *Computers* **2019**, *8*, 21. [CrossRef]
50. Azab, A.; Khasawneh, M.; Alrabaee, S.; Choo, K.K.R.; Sarsour, M. Network traffic classification: Techniques, datasets, and challenges. *Digit. Commun. Netw.* **2024**, *10*, 676–692. [CrossRef]